BAB 2
TINJAUAN PUSTAKA

2.1 Profil Instansi

2.1.1 Sejarah Instansi

CV.Sultan Jaya Mandiri yang berlokasi di Jalan Raya Nanggeleng No.53 Darangdan-Purwakarta, bergerak dalam bidang penjualan barang. Barang-barang yang dijual terdiri dari barang-barang kebutuhan sehari-hari seperti sembako, food, dan non-food.

Toko Sultan pertama kali berdiri pada tanggal 18 Agustus 2005, toko sultan menerapkan konsep swalayan dalam proses penjualannya, sehingga memudahkan konsumen untuk melihat dan memilih langsung produk yang akan dibeli. Toko sultan pada awalnya hanya menjual produk-produk secara retail, namun seiring meningkatnya omset penjualan maka toko sultan menjual produk dalam jumlah yang banyak atau grosir. Selain menjual produk dalam bentuk satuan, toko sultan pun menjual produk dalam bentuk paket dengan harga jual lebih murah dibandingkan dengan membeli produk secara terpisah atau satuan. Pada tanggal 22 September 2014 Toko Sultan resmi menjadi badan usaha yang terdaftar dengan nama CV.Sultan Jaya Mandiri. Dan dari situlah Toko sultan mulai berkembang dan menjadi perusahaan distributor barang seperti saat ini.

2.1.2 Logo

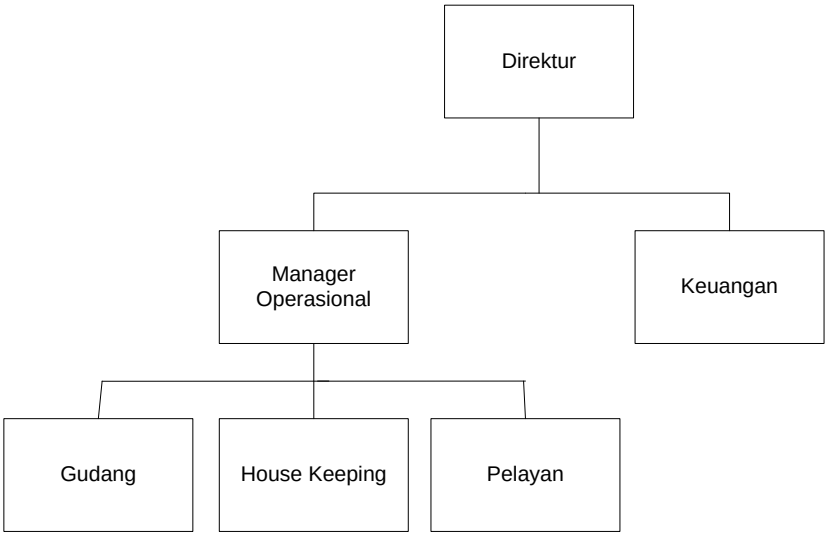
Berikut adalah logo dari CV.Sultan Jaya Mandiri yang dapat dilihat pada Gambar 2.1:



Gambar 2. 1 Logo CV.Sultan Jaya Mandir

1.1.3 Struktur Organisasi

Struktur organisasi instansi adalah gambar yang menerangkan posisi dan hierarki kerja di dalam suatu perusahaan. Struktur organisasi CV.Sultan Jaya Mandiri dapat dilihat pada Gambar 2.2:



Gambar 2. 2 Struktur Organisasi CV.Sultan Jaya Mandiri

Job Description

1. Direktur bertanggung jawab penuh terhadap perusahaan, tugas dan tanggung jawabnya sebagai berikut :
   a. Membuat perencanaan, strategi dan kebijakan yang menyangkut operasi Perusahaan.
   b. Menyusun anggaran kebutuhan persediaan produk.
   c. Melakukan kontrol secara keseluruhan atau operasi Perusahaan.
   d. Memegang kendali atas keputusan penting yang bersifat umum berkaitan dengan keuangan.
2. Manager Operasional berfungsi untuk membantu Direktur, tugas dan tanggung jawabnya adalah sebagai berikut :
   a. Membantu Direktur dalam mengawasi Perusahaan.
   b. Mengatur setiap bagian yang ada di Perusahaan agar menjalankan tugasnya dengan baik.

3. Bagian Keuangan bertugas terhadap transaksi pelanggan, tugas dan tanggung jawab sebagai berikut :
   a. Bertanggung jawab terhadap hal-hal yang menyangkut keuangan.
   b. Menghitung pemasukan dan pengeluaran setiap bulannya.
   c. Melayani pelanggan dalam pembayaran transaksi.
4. Bagian Gudang bertugas terhadap persediaan barang digudang, tugas dan tanggung jawabnya sebagai berikut:
   a. Bertanggung jawab terhadap hal-hal yang menyangkut persediaan barang.
   b. Mengecek barang yang kosong.
   c. Mengecek barang kadaluarsa.
5. Bagian Pelayan berfungsi untuk melayani setiap pelanggan, tugas dan tanggung jawab sebagai berikut :
   a. Membantu dalam melayani pesanan-pesanan pelanggan.
   b. Melayani semua pesanan pelanggan.
6. Bagian *House Keeping* bertugas terhadap kebersihan dan kenyamanan pelanggan, tugas dan tanggung jawab sebagai berikut :
   a. Bertanggung jawab terhadap kebersihan, baik di luar ataupun di dalam lingkungan Perusahaan/Toko.
   b. Membereskan dan membersihkan peralatan yang telah digunakan.

2.1.3 Visi dan Misi Instansi

Visi yang dijalankan perusahaan adalah menjadi perusahaan distributor lokal yang berkepanjangan. Misi perusahaan ada 2 jenis yaitu:

1. Produktif

   Meningkatkan produktifitas perusahaan dan sumber daya manusianya agar perusahaan dapat terus berkembang kearah yang lebih baik.
2. Efisiensi

   Perusahaan harus selalu efisien dalam setiap mengerjakan kegiatannya agar konsumen dapat terlayani dengan baik.

2.2 Landasan Teori

Landasan teori membahas mengenai materi atau teori apa saja yang digunakan sebagai acuan dalam membuat tugas akhir ini. Landasan teori yang di uraikan merupakan hasil studi literature, baik dari buku, maupun situs internet.

2.2.1 Data

Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, symbol teks, gambar, bunyi, atau kombinasinya [5]. Dalam pendekatan basis data tidak hanya berisi basis data itu sendiri tetapi juga termasuk definisi atau deskripsi dari data yang disimpan. Definisi data disimpan dalam sistem katalog, yang berisi informasi tentang struktur tiap berkas, tipe dan format penyimpanan tiap item data, dan berbagai konstrin dari data. Semua informasi yang disimpan dalam katalog ini biasa disebut *meta-data* [6].

2.2.2 Basis Data

Basis data adalah mekanisme yang digunakan untuk menyimpan informasi atau data. Informasi adalah sesuatu yang kita gunakan sehari-hari untuk berbagai alasan. Dengan basis data, pengguna dapat menyimpan data secara terorganisasi. Setelah data disimpan, informasi harus mudah diambil. Kriteria dapat digunakan untuk mengambil informasi. Cara data disimpan dalam basis data menentukan seberapa mudah mencari informasi berdasarkan banyak kriteria. Data pun harus mudah ditambahkan ke dalam basis data, dimodifikasi, dan dihapus [7].

Menurut Fathansyah [5], basis data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.

3. Kumpulan file/ tabel/ arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

2.2.2.1 Data pada Basis Data dan Hubungannya

Ada 3 jenis data pada sistem *database*, yaitu [8] :

1. Data operasional dari suatu organisasi, berupa *data* yang tersimpan dalam basis data.
2. Data masukan (*input data*), data dari luar sistem yang dimasukan melalui peralatan input (*keyboard*), yang dapat merubah data operasional.
3. Data kelauran (*output data*), berupa laporan melalui peralatan *output* sebagai hasil dari dalam sistem yang mengakses data operasional.

2.2.2.2 Keuntungan dan Kerugian Pemakain Sistem Database

Keuntungan [8] :

1. Terpeliharanya keselarasan *data*
2. *Data* dapat dipakai secara bersama-sama.
3. Memudahkan penerapan standarisaasi dan batas-batas pengamanan.
4. Terpeliharanya keseimbangan atas perbedaan kebutuhan data dari setiap aplikasi.
5. Program /data *independent*.

Kerugian [8]:

1. Mahal dalam implementasinya.
2. Rumit
3. Penanganan proses *recovery backup* sulit.
4. Kerusakan pada sistem basis data dapat mempengaruhi.

2.2.3 Database Management System

Kumpulan atau gabungan *database* dengan perangkat lunak aplikasi yang berbasis *database* tersebut dinamakan *Database Management System (DBMS). DBMS* merupakan koleksi terpadu dari *database* dan program–program komputer (utilitas) yang digunakan untuk mengakses dan memelihara *database*. Program-program tersebut menyediakan berbagai fasilitas operasi untuk memasukan, melacak, dan memodifikasi data kedalam *database*, mendefinisikan *data* baru,

serta mengolah data menjadi informasi yang dibutuhkan (*DBMS = Database* + Program Utilitas) [5].

2.2.4 Data Mining

Data mining, sering juga disebut *knowledge discovery in database* (KDD), adalah kegiatan yang meliputi pengumpulan, pemakaian data historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar. Keluaran dari data mining ini bisa dipakai untuk memperbaiki pengambilan keputusan di masa depan. Sehingga istilah *pattern recognition* sekarang jarang digunakan karena ia termasuk bagian dari *data mining* [1].

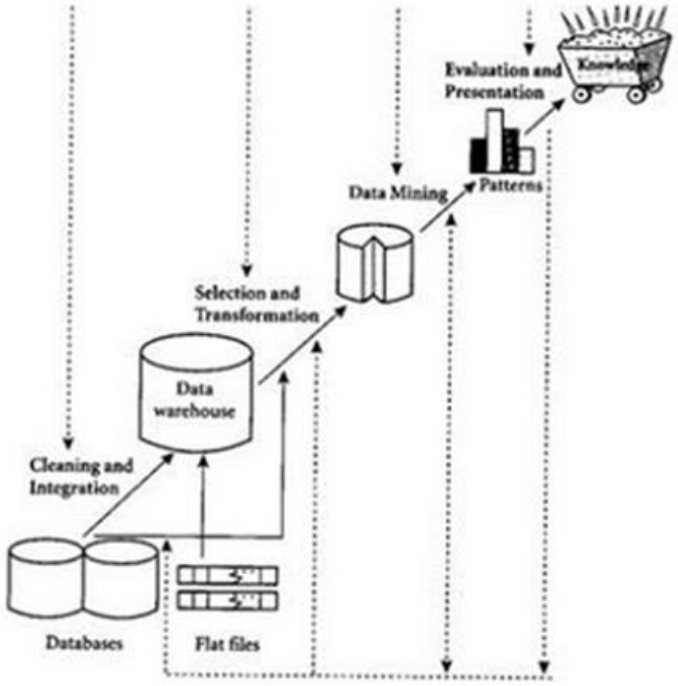
Secara umum, definisi *data mining* dapat diartikan sebagai berikut [9]:

1. Proses penemuan pola yang menarik dari data yang tersimpan dalam jumlah besar.
2. Ekstrasi dari suatu informasi yang berguna atau menarik (non-trivial, implisit, sebelumnya belum diketahui potensi kegunaannya) pola atau pengetahuan dari data yang di simpan dalam jumlah besar.
3. Eksplorasi dari analisa secara otomatis atau semiotomatis terhadap data-data dalam jumlah besar untuk mencari pola dan aturan yang berarti.

2.2.4.1 Tahapan-tahapan Data Mining

Data mining sangat diperlukan terutama dalam mengelola *data* yang sangat besar untuk memudahkan aktifitas *recording* suatu transaksi dan untuk proses *data warehousing* agar dapat memberikan informasi yang akurat bagi pengguna *data mining*. Alasan utama *data mining* sangat dibutuhkan dalam industri informasi karena tersedianya data dalam jumlah yang besar dan semakin besarnya kebutuhan untuk mengubah data tersebut menjadi informasi dan pengetahuan yang berguna karena sesuai fokus bidang ilmu ini yaitu melakukan kegiatan mengekstraksi atau menambang pengetahuan dari data yang berukuran atau berjumlah besar. Informasi iniliah yang nantinya sangat berguna untuk pengembangan. Berikut adalah langkah-langkah dalam data mining [1].

Tahapan-tahapan proses data mining tersebut dapat dilihat pada Gambar 2.3:



Gambar 2. 3 Tahapan Proses Data Mining

Tahapan-tahapan data mining :

1. Pembersihan data (*data cleaning*)

   Pembersihan data merupakan proses menghilangkan noise dan data yang tidak konsisten. Pembersihan data ini juga akan mempengaruhi performasi dari teknik *data mining* karena data yang ditangani akan berkurang jumlah dan kompleksitasnya.

2. Integrasi data (*data integration*)

   Integrasi data merupakan penggabungan data dari berbagai *database* ke dalam satu *database* baru. Tidak jarang data yang diperlukan untuk *data mining* tidak hanya berasal dari satu *database* saja, tetapi bisa juga berasal dari beberapa *database* atau *file* teks. Integrasi data perlu dilakukan secara teliti dan cermat karena kesalahan pada integrasi data bisa menghasilkan hasil yang menyimpang dan dapat menyesatkan proses pengambilan aksi nantinya.

3. Seleski data (*data selection*)

   Data yang ada pada suatu *database* acap kali tidak semua terpakai, oleh karena itu hanya data yang sesuai kebutuhan saja yang akan dipakai.

Sebagai contoh, dalam mencari nilai asosiasi suatu barang, data nama pelanggan dalam data transaksi tidak diperlukan, karena data nama pelanggan tersebut tidak diperlukan, data tersebut dapat dihilangkan.

4. Transformasi data (*data transformation*)

   Data diubah atau digabung ke dalam format yang sesuai untuk diproses dalam data mining. Beberapa metode data mining membutuhkan suatu format data khusus sebelum bisa diaplikasikan.

5. Proses mining

   Proses utama dalam tahapan data mining, disaat inilah metode-metode yang digunakan diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data.

6. Evaluasi pola (*pattern evaluation*)

   Untuk mengidentifikasi pola-pola menarik kedalam knowledge based yang ditemukan. Dalam tahap ini hasil dari teknik data mining berupa pola-pola yang khas maupun model prediksi dievaluasi untuk menilai apakah hipotesa yang ada memang tercapai. Bila ternyata hasil yang didapat tidak sesuai, maka ada beberapa *alternative* yang dapat diambil, seperti menjadikannya umpan balik untuk memperbaiki proses data mining atau mencoba metode *data mining* lainnya yang lebih sesuai.

7. Presentasi pengetahuan (*knowledge presentation*)

   Tahap terakhir dari proses data mining ini adalah bagaimana memformulasikan keputusan atau aksi dari hasil data mining yang didapat sehingga dapat dipahami oleh semua pihak.

2.2.4.2 Metode - Metode Data Mining

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu [10] :

1. Deskripsi

   Terkadang peneliti dan analis secara sederhana ingin mencoba mencari cara untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat mengumpulkan keterangan atau fakta bahwa siapa yang tidak cukup

profesional akan sedikit didukung dalam pemilihan presiden. deskripsi dari pola dan kecenderungan sering memberikan kemungkinan penjelesan untuk suatu pola atau kecenderungan.

2. Estimasi

   Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih kearah numerik daripada ke arah kategori. Model dibangun dengan record lengkap yang menyediakan nilai dari variabel target sebagai nilai prediksi. Sebagai contoh, akan dilakukan estimasi tekanan darah sistolik pada pasien rumah sakit berdasarkan umur pasien, jenis kelamin, indeks berat badan, dan level sodium darah. Hubungan antara tekanan darah sistolik dan nilai variabel prediksi dalam proses pembelajaran akan menghasilkan model estimasi. Model estimasi yang dihasilkan dapat digunakan untuk kasus baru lainnya.

3. Prediksi

   Prediksi hampir sama dengan klasifikasi dan estimasi, kecuali bahwa dalam prediksi nilai dari hasil akan ada dimasa mendatang. Contoh prediksi dalam bisnis dan penelitian adalah :

   1. Prediksi harga beras dalam tiga bulan yang akan datang.
   2. Prediksi persentase kenaikan kecelakaan lalu lintas tahun depan jika batas bawah dinaikan.

4. Klasifikasi

   Dalam klasifikasi, terdapat terget variabel kategori. sebagai contoh, penggolongan pendapatan dapat dipisahkan dalam tiga kategori yaitu: pendapatan tinggi, pendapatan sedang, dan pendapatan rendah. Contoh lain klasifikasi dalam bisnis dan penelitian adalah :

   1. Menentukan apakah suatu transaksi kartu kredit merupakan transaksi yang curang atau bukan.
   2. Mendiagnosis penyakit seorang pasien untuk mendapatkan termasuk kategori penyakit apa.

5. Pengklusteran

   Pengklusteran merupakan pengelompokan *record*, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan.

   Kluster adalah kumpulan *record* yang memiliki kemiripan satu dengan yang lainnya dan tidak memiliki kemiripan dengan *record-record* dalam kluster lain. Pengklusteran berbeda dengan klasifikasi yaitu tidak adanya *variabel* target dalam pengklusteran. pengklusteran tidak mencoba untuk melakukan klasifikasi, mengestimasi, atau memprediksi nilai dari *variabel target*. Akan tetapi, *algoritma* pengklusteran mencoba untuk melakukan pembagian terhadap keselurahan data menjadi kelompok-kelompok yang memiliki kemiripan (*homogen*), yang mana kemiripan *record* dalam suatu kelompok akan bernilai maksimal, sedangkan kemiripan dengan *record* dalam kelompok lain akan bernilai minimal. Contoh pengklusteran dalam bisnis dan penelitian adalah :

   1. Mendapatkan kelompok-kelompok konsumen untuk target pemasaran dari suatu produk sebuah perusahaan yang tidak memiliki dana pemesaran yang besar.
   2. Untuk tujuan audit akuntansi, yaitu melakukan pemisahan terhadap perilaku finansial dalam baik maupun mencurigakan.

6. Asosiasi

   Tugas asosiasi dalam *data mining* adalah menemukan atribut yang muncul dalam satu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang pasar. Contoh asosiasi dalam bisnis dan penelitian adalah :

   1. Meneliti jumlah pelanggan dari perusahaan telekomunikasi seluler yang diharapkan untuk memberika respon positif terhadap penawaran *upgrade* layanan yang diberikan.
   2. Menemukan barang dalam supermarket yang dibeli secara bersamaan dan barang yang tidak pernah dibeli secara bersamaan.

2.2.5 *Association Rule*

Aturan asosiasi *(Association rule)* adalah salah satu teknik tentang ‘apa bersama apa’. Ini bisa berupa transaksi di supermarket, misalnya seseorang yang membeli susu bayi juga membeli sabun mandi. Di sini berarti susu bayi bersama sabun mandi. Karena awalnya berasal dari studi tentang *database* transaksi pelanggan untuk menentukan kebiasaan suatu produk dibeli bersama produk apa, maka aturan asosiasi juga sering dinamakan *Market Basket* [5].

Untuk mencari *association rule* dari suatu kumpulan data, tahap pertama yang harus dilakukan adalah mencari *frequent itemset* terlebih dahulu. *Frequent itemset* adalah sekumpulan *item* yang sering muncul secara bersamaan. Setelah semua pola *frequent itemset* ditemukan, barulah mencari aturan asosiatif atau aturan keterkaitan yang memenuhui syarat yang telah ditentukan.

Jika diasumsikan bahwa barang yang dijual di swalayan adalah semesta, maka setiap barang akan memiliki *Boolean variable* yang akan menunjukan keberadaannya atau tidak barang tersebut dalam satu transaksi atau satu keranjang belanja. Pola *Boolean* yang didapat digunakan untuk menganalisa barang yang dibeli secara bersamaan. Pola tersebut dirumuskan dalam sebuah *association rule.* Sebagai contoh konsumen biasanya akan membeli kopi dan susu yang ditujukan sebagai berikut :

Kopi → susu [*support* = 2%, *confidence* = 60%]

Nilai *support* 2% menunjukan bahwa keseluruhan dari total transaksi konsumen membeli kopi dan susu secara bersamaan yaitu sebanyak 2%. Sedangkan *confidence* 60% yaitu menunjukan bila konsumen membeli kopi dan pasti membeli susu sebesar 60%.

Penting tidaknya suatu aturan asosiatif dapat diketahui dengan dua parameter, yaitu *support* dan *confidence. Support* (nilai penunjang) adalah presentase kombinasi *item* tersebut dalam *database*, sedangkan *confidence* (nilai kepastian) adalah kuatnya hubungan antar-*item* dalam aturan asosiasi.

Dalam menentukan suatu *association rule*, terdapat suatu ukuran kepercayaan yang didapat dari hasil pengelolahan data dengan perhitungan tertentu. Umumnya ada dua ukuran, yaitu:

a. *Support* : suatu ukuran yang menunjukan seberapa besar tingkat dominasi suatu *itemset* dari keseluruhan transaksi. Ukuran ini menentukan apakah suatu *itemset* layak untuk dicari *confidence*-nya (misal, dari keseluruhan transaksi yang ada, seberapa besar tingkat dominasi suatu *item* yang menunjukan bahwa *item* A dan *item* B dibeli bersamaan).
b. *Confidence* : suatu ukuran yang menunjukan hubungan antara 2 *item* secara *conditional* (misal, menghitung kemungkinan seberapa sering *item* B dibeli oleh pelanggan jika pelanggan tersebut membeli sebuah *item* A).

Kedua ukuran ini nantinya berguna dalam menentukan kekuatan suatu pola dengan membandingkan pola tersebut dengan nilai minimum kedua parameter tersebut yang ditentukan oleh pengguna. Analisis asosiasi didefinisikan suatu proses untuk menemukan semua aturan asosiasi yang memenuhi syarat minimum untuk *support* (*minimum support*) dan syarat minimum untuk *confidence* (*minimum confidence*).

Nilai *support* sebuah *item* diperoleh dengan rumus sebagai berikut[1] :

$$Support(\mathrm{A}) = \frac{Jumlah\ transaksi\ mengandung\ A}{Total\ Transaksi} \mathrm{x}100\% \ldots\ldots \text{Persamaan (2.1)}$$

Sementara itu, nilai *support* dari 2 *item* diperoleh dari rumus berikut:

$$Support\ (\mathrm{A}, \mathrm{B}) = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Jumlah transaksi}} \mathrm{x}100\% \text{..Persamaan (2.2)}$$

Sedangkan nilai *confidence* dapat dicari setelah pola frekuensi munculnya sebuah *item* ditemukan. Rumus untuk menghitung *confidence* adalah sebagai berikut:

$$Confidence = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Jumlah transaksi yang mengandung A}} \mathrm{x}100\% \ldots\ldots \text{Persamaan(2.3)}$$

Berikut adalah contoh penerapan metode *association rule* :

Misalkan diberikan tabel data transaksi sebagai berikut dengan minimum *support count* = 2.

Tabel 2. 1 Data Transaksi

TID	*Items*
1	B
2	B, D
3	A, C, D, E

TID	*Items*
4	A, D, E
5	B, C
6	A, C
7	A

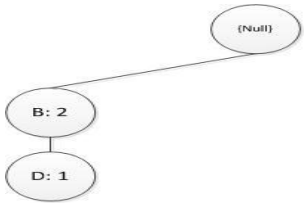
Yang mana nilai *support count* didapat dari : $\frac{15}{7}$ x 100% = 2.14 ≈ 2

Frekuensi kemunculan tiap item dapat dilihat pada tabel berikut:

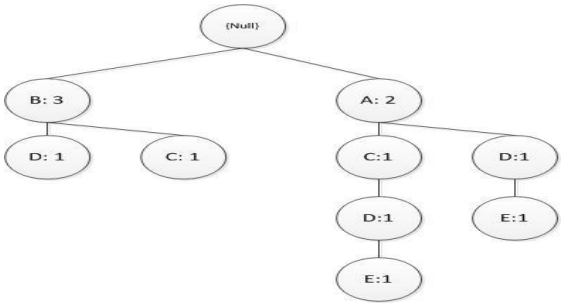
Tabel 2. 2 Frekuensi Kemunculan Tiap Karakter

Item	Frekuensi
A	4
B	3
C	3
D	3
E	2

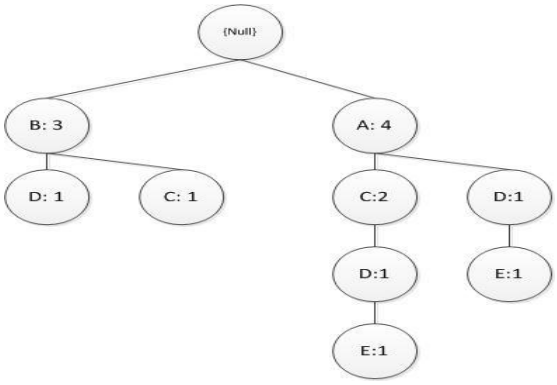
Karena contoh kasus dalam kemunculan *item* sudah *frequent* dalam setiap transaksi, sudah diurut berdasarkan frekuensi yang paling tinggi. Selanjutnya pada gambar di bawah ini memberikan ilustrasi mengenai pembentukan FP-Tree setelah pembacaan TID 2



Gambar 2. 4 Hasil Pembentukan FP-Tree setelah pembacaan TID 2



Gambar 2. 5 Hasil Pembentukan FP-Tree setelah pembacaan TID 5



Gambar 2. 6 Hasil pembentukan FP-Tree setelah pembacaan TID 7

Diberikan 7 data transaksi dengan 5 jenis item seperti pada tabel di atas. Gambar 5-7 menunjukan proses terbentuknya FP-Tree setiap TID dibaca. Setiap simpul pada FP-Tree mengandung nama sebuah item dan *counter support* yang berfungsi untuk menghitung frekuensi kemunculan item tersebut dalam tiap lintasan transaksi.

Setelah FP-Tree terbentuk dari sekumpulan data transaksi maka, proses pencarian *frequent itemset* dengan menggunakan algoritma FP-Growth akan dilakukan yang dibagi menjadi tiga tahapan utama, yaitu:

a. Tahap pembangkitan *conditional pattern base*

   Merupakan subdata yang berisi *prefix path* (lintasan awal) dan *suffix pattern* (pola akhiran). Pembangkitan *conditional pattren base* didapatkan melalui FP-Tree yang telah dibangun sebelumnya.

b. Tahap pembangkitan *conditional* FP-Tree

   Pada tahap ini, *support count* dari setiap *item* pada setiap *conditional pattern base* dijumlahkan, lalu setiap *item* yang memiliki jumlah *support count* lebih besar atau sama dengan minimum *support count* akan dibangkitkan dengan *conditional* FP-Tree seperti pada tabel 2.3 dibawah ini.

Tabel 2. 3 Hasil Perhitungan Conditional Pattern Base dan Conditional FP-Tree

Item	*Conditional Pattern Base*	*Conditional* FP Tree
E	{{A,C,D:1}, {A,D:1}}	{<A:2,D:2>}
D	{{B:1}, {A,C:1}, {A:1}}	{A:2}
C	{B:1}, {A:2}	{A:2}

c. Tahap pencarian *frequent itemset.*

Apabila *conditional* FP-Tree merupakan lintasan tunggal(*single-path*), maka didapatkan *frequent itemset* dengan melakukan kombinasi *item* untuk setiap *conditional* FP-Tree. Jika bukan lintasan tunggal, maka dilakukan pembangkitan FP-Growth secara *rekursif* (proses memanggil dirinya sendiri) seperti contoh pada tabel 2.4 dibawah ini.

Tabel 2. 4 Hasil Frequent Itemset

Suffix	*Frequent Itemset*
E	E, A-E, D-E, A-D-E
D	D, A-D
C	C, A-C

Tabel 2. 5 Hasil Generate Rule

Rule	*Support*	*Confidence* (%)
A → E	2	(2/4)*100 = 50
E → A	2	(2/2)*100 = 100
A → D	2	(2/4)*100 = 50
D → A	2	(2/3)*100 = 66.67
A → C	2	(2/4)*100 = 50
C → A	2	(2/3)*100 = 66.67
D → E	2	(2/3)*100 = 66.67
E → D	2	(2/2)*100 = 100
A- D →E	2	(2/2)*100 = 100
D- E →A	2	(2/2)*100 = 100

2.2.6 Algoritma FP- Growth

Algoritma yang sama dengan Apriori, *FP-Growth* mulai dengan menghitung item tunggal sesuai dengan jumlah kemunculan *item* yang ada didalam *dataset*. Setelah proses penghitungan selesai maka akan dibuat struktur pohon pada tahap kedua. Pohon yang dibuat mulanya kosong yang nanti akan

diisi dengan hasil dari dataset yang telah didapat sebelumnya. Kunci untuk mendapatkan struktur pohon yang bisa didapatkan dengan proses lebih cepat untuk mencari item set yang besar menjadi sedikit dengan di urutkan secara *descending* dari frekuensi yang ada *dataset* tersebut. Masing-masing item yang tidak mencapai kebutuhan minimum dari *threshold* tidak dimasukkan kedalam pohon, tapi dikeluarkan secara efektif dari *dataset* [14].

FP-Tree merupakan struktur penyimpanan data yang dimampatkan. *FP-Tree* dibangun dengan memetakan setiap data transaksi ke dalam setiap lintasan tertentu dalam *FP-Tree*. Karena dalam setiap transaksi yang dipetakan, mungkin ada transaksi yang memiliki *item* yang sama, maka lintasannya memungkinkan untuk saling menimpa. Semakin banyak data transaksi yang memiliki *item* yang sama, maka proses pemampatan dengan struktur data *FP-Tree* semakin efektif. Kelebihan dari *FP-Tree* adalah hanya memerlukan dua kali pemindaian data transaksi yang terbukti sangat efisien.

Adapun *FP-Tree* adalah sebuah pohon dengan definisi sebagai berikut:

a. *FP-Tree* dibentuk oleh sebuah akar yang diberi label *null*, sekumpulan berupa pohon yang beranggotakan item-item tertentu, dan sebuah tabel *frequent header*.
b. Setiap simpul dalam *FP- Tree* mengandung tiga informasi penting, yaitu label *item*, menginformasikan jenis *item* yang direpresentasikan simpul tersebut, *support count*, merepresentasikan jumlah lintasan transaksi yang melalui simpul tesebut, dan *pointer* penghubung yang menghubungkan simpul-simpul dengan label item sama antar-lintasan, ditandai dengan garis panah putus-putus.

2.2.4.1 Langkah-Langkah Proses Perhitungan *Association Rule* Dengan *Algoritma FP-Growth*

Proses perhitungan *association rule* terdiri dari beberapa tahap adalah sebagai berikut [15] :

1. Membuat *Header Item*

   Header dalam hal ini selain sebagai *header* suatu *item* ke FP-Tree juga sebagai jenis *item* dasar yang memenuhi minimum *support*. Setelah

mendapatkan *item* dan nilai *support*-nya, maka *item* yang tidak *frequent* dibuang dan *item* diurutkan berdasarkan nilai *support*-nya. Header untuk item, disiapkan pada suatu *array* tertentu dan ditambahkan ketika membuat FP-Tree.

2. Membuat *FP-Tree*

   FP-Tree dibangun dengan mencari *item* sesuai urutan pada *item* yang *frequent*. Data transaksi tidak perlu diurutkan, dan untuk tiap *item* yang ditemukan bisa langsung dimasukkan ke dalam *FP-Tree.* Sesudah membuat *root*, tiap *item* yang ditemukan dimasukkan berdasarkan *path* pada *FP-Tree.* Jika *item* yang ditemukan sudah ada, maka nilai *support item* tersebut yang ditambahkan. Namun jika *path* belum ada, maka dibuat *node* baru untuk melengkapi *path* baru pada *FP-Tree* tersebut. Hal ini dilakukan selama item pada transaksi masih ada yang *qualified*, artinya memenuhi nilai minimum *support*. Jadi, *item-item* yang ditemukan dalam transaksi akan berurutan memanjang ke bawah. Dalam struktur *FP-Tree*, diterapkan alur *path* dari *child* hingga ke *root*. Jadi, suatu *path* utuh dalam *FP-Tree* adalah dari *child* terbawah hingga ke *root*. Tiap *node* pada *FP-Tree* memiliki *pointer* ke *parent*, sehingga pencarian harus dimulai dari bawah.

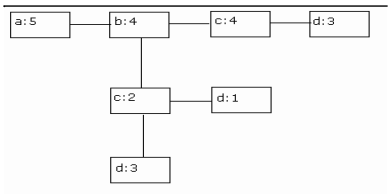
3. *Pattern Extraction*

   Pattern extraction dilakukan berdasarkan keterlibatan *item* pada suatu *path*. Di setiap *path*, diperiksa semua kombinasi yang mungkin dimana *item* tersebut terlibat. Di iterasi berikutnya dilakukan dengan melibatkan *item* berikutnya, tanpa melibatkan *item* sebelumnya, sehingga *pattern* yang sama tidak akan ditemukan dua kali pada *path* yang sama. Bila *item* pertama suatu hasil kombinasi bukan *item* terakhir (sebelum *root*), maka kombinasi *itemset* tersebut masih bisa dikembangkan lagi.

4. Memasukkan setiap *pattern* yang ditemukan ke dalam *PatternTree*

   Setelah mengolah *FP-Tree* menjadi *pattern-pattern*, diperlukan proses akumulasi *pattern-pattern* yang ditemukan mengingat *pattern* yang

sama dapat ditemukan pada path yang berbeda. Untuk itu digunakan struktur data *Pattern Tree* (lihat Gambar 2.5). Setiap node di *Pattern Tree* merepresentasikan dan menyimpan frekuensi suatu *pattern*. *Pattern Tree* terdiri atas *Pattern TreeNode* yang menyimpan nilai item, nilai *support* dan dilengkapi dengan dua *pointer* yaitu untuk horisontal dan vertikal.



Gambar 2. 7 Hasil *Generate Rule*

Misalnya pada *node* d:1 di atas, berarti terdapat *pattern* a-c-d bernilai *support* 1. Kemudian bila ada pattern a-c-d lagi bernilai *support* n yang ditemukan dari *FP-Tree* maka nilai *support* 1 tersebut menjadi n+1. Contoh hasil lengkap dari *PatternTree* tersebut:

1. a:5 menggambarkan bahwa ada pattern a sebanyak 5
2. b:4 menggambarkan bahwa ada pattern a-b sebanyak 4
3. c:4 menggambarkan bahwa ada *pattern* a-b-c sebanyak 4
4. d:3 menggambarkan bahwa ada *pattern* a-b-c-d sebanyak 3
5. c:2 menggambarkan bahwa ada *pattern* a-c sebanyak 2
6. d:1 menggambarkan bahwa ada *pattern* a-c-d sebanyak 1
7. d:3 menggambarkan bahwa ada *pattern* a-d sebanyak 3

5. Mengurutkan dan Menyeleksi *Pattern*

   Pattern yang tidak memenuhi minimum *support*, dihapus dari daftar *pattern*. *Pattern-pattern* yang tersisa kemudian diurutkan untuk memudahkan pembuatan *rules*.

2.2.7 *Inified Modelling Language* (UML)

Unified Modeling Language (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek serta aplikasinya. Berikut adalah beberapa model yang digunakan dalam perancangan Data Mining

Pemaketan Produk di CV.Sultan Jaya Mandiri untuk menggambarkan sistem dalam UML:

1. Diagram *Use Case*
2. Diagram Kelas
3. Diagram Aktivitas
4. Diagram Objek

2.2.7.1 Diagram Use Case

Diagram use case adalah model fungsional sebuah sistem yang menggunakan aktor dan *use case*. *Use case* adalah layanan (s*ervices*) atau fungsi–fungsi yang disediakan oleh sistem untuk penggunanya.

Deskripsi Diagram Use Case:

1. Sebuah *use case* adalah dimana sistem digunakan untuk memenuhi satu atau lebih kebutuhan pemakai.
2. *Use case* merupakan awal yang sangat baik untuk setiap fase pengembangan berbasis objek, design testing, dan dokumentasi.
3. *Use case* menggambarkan kebutuhan sistem dari sudut pandang di luar sistem.
4. Use case menentukan nilai yang diberikan sistem kepada pemakainya.
5. *Use case* hanya menetapkan apa yang seharusnya dikerjakan oleh sistem, yaitu kebutuhan fungsional sistem.
6. *Use case* tidak untuk menentukan kebutuhan nonfungsional, misal: sasaran kerja, bahasa pemrograman.

2.2.7.2 Diagram Kelas

Diagram kelas adalah diagram UML yang menggambarkan kelas-kelas dalam sebuah sistem dan hubungannya antara satu dengan yang lain, serta dimasukkan pula atribut dan operasi. Tahapan dari diagram kelas adalah sebagai berikut:

1. Mengidentifikasi objek dan mendapatkan kelas-kelasnya.
2. Mengidentifikasi atribut kelas-kelas.
3. Mulai mengkonstruksikan kamus data.

4. Mengidentifikasi operasi pada kelas-kelas.
5. Mengidentifikasikan hubungan antar kelas dengan menggunakan *asosiasi*, *agregasi*, dan *inheritance* (pewarisan).

2.2.7.3 Diagram Aktifitas

Diagram aktivitas adalah representasi grafis dari seluruh tahapan alur kerja. Diagram ini mengandung aktivitas, pilihan tindakan, perulangan dan hasil dari aktivitas tersebut[14]. Diagram ini dapat digunakan untuk menjelaskan proses bisnis dan alur kerja operasional secara langkah demi langkah dari komponen suatu sistem.

2.2.7.4 Diagram Objek

Diagram objek adalah suatu diagram yang berfungsi untuk mengatur atribut, objek dan hubungan antara contoh dalam diagram[14]. Diagram objek juga dapat menampilkan struktur model sistem dalam waktu tertentu. Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek harus dipastikan semua kelas yang sudah didefinisikan pada diagram kelas harus dipakai objeknya, karena jika tidak, pendefinisian kelas itu tidak dapat dipertanggungjawabkan. Meskipun diagram kelas diperlukan untuk mendokumentasikan struktur kelas, tetapi diagram objek, dapat berguna. Sebuah diagram objek dasarnya instansiasi semua atau bagian dari diagram kelas.